Computational Geometry: Theory and Applications 3 (1993) 139–155
Elsevier

COMGEO 150

A competitive analysis of algorithms for searching unknown scenes

Bala Kalyanasundaram* and Kirk Pruhs**
Computer Science Department, University of Pittsburgh, Pittsburgh, PA 15260, USA

Communicated by Paul Spirakis
Submitted 24 September 1991
Accepted 4 March 1993

Abstract

We consider problems involving robot motion planning in an initially unknown scene of obstacles. The two specific problems that we examine are mapping the scene and searching the scene for a recognizable target whose location is unknown. We use competitive analysis as a tool for comparing algorithms. In the case of convex polygonal obstacles, we show a tight $\Theta(\min(k, \sqrt{k\bar{\alpha}}))$ bound on the competitive factor for these problems, where $\bar{\alpha}$ and k are the average aspect ratio and the number of objects, respectively. We also consider the competitive factor for the Nearest Neighbor heuristic.

1. Introduction

1.1. Problem statement

The setting for the problems that we consider is a Euclidean plane sprinkled with k obstacles. Inhabiting this environment is a robot that is equipped with a vision sensor. The robot learns about the objects and the environment only through the visual information provided by the sensor. More precisely, the robot is only aware of parts of objects that lie within its line of sight currently, or that lied within its line of sight at some point in the past. For example, in Fig. 1 the

Correspondence to: Bala Kalyanasundaram, Computer Science Department, University of Pittsburgh, Pittsburgh, PA 15260, USA.

* Supported in part by National Science Foundation CCR-9009318 and CCR-9202158.
** Supported in part by National Science Foundation CCR-9209283.

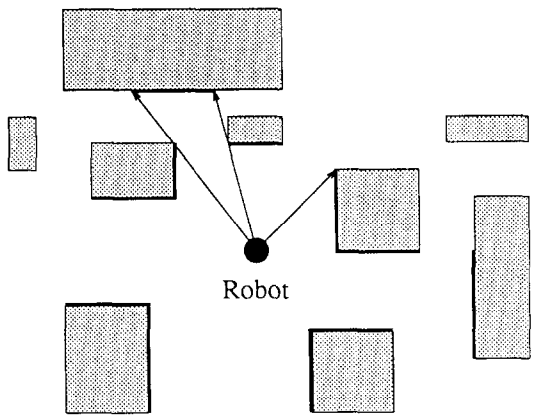

Fig. 1.

robot can only see the parts of the objects that are darkened. The goal of the robot is move through the scene, avoiding obstacles, until it accomplishes some goal. The two major goals that we consider are the following.

Searching. In the searching problem the robot must find the location of some recognizable object, called the *target* in the scene. Imagine that the robot is looking for a McDonalds restaurant in an unknown city. The robot may not know the location of the McDonalds, but it can certainly recognize the golden arches once it sees them. More precisely, the robot must move along an obstacle-avoiding path P terminating at the target. Alternatively, one might only require the robot to move along an obstacle avoiding path P terminating at point from which some part of the target may be seen. All of our bounds also hold for this variant of searching.

Mapping. In the mapping problem the robot's goal is to determine the location of all of the obstacles. More precisely, the robot must move along an obstacle avoiding path P that has the property that there is an obstacle avoiding straight line path from every point on the boundary of each obstacle to some point in P. As we shall see the difficulty of the problem is unchanged even if the robot is required to see only a part of each object, instead of the entire perimeter.

In each of these problems we want to minimize the length of the robot's path P. Since the robot is learning the environment as it moves, we can view these problems as online problems. In this paper we will use competitive analysis to compare different online algorithms for these problems. Let OPT be the shortest path that would allow the robot to accomplish its goal. We can assume that OPT is computed by an offline algorithm that has a bird's eye view of the environment. A robot's algorithm has a *competitive factor* of c for a particular scene S if the

ratio of the length of P divided by the length of OPT is c. The competitive factor of robot's algorithm with respect to a set $\mathcal{S}$ of scenes is just the supremum, over scenes $S \in \mathcal{S}$, of the competitive factor for S. To say that an algorithm A is *c-competitive* is equivalent to saying that A has a competitive factor of c.

For us a good online algorithm for the robot is one with a minimal competitive factor. However, it also important that the online algorithms be based on simple heuristics, as well as have low competitive factors. We will thus also analyze the competitive factor of some simple heuristics. For the problems that we consider the advantage of competitive analysis, versus worst case analysis, is that competitive analysis is a more expressive measure for the complexity of the online algorithms.

1.2. Previous results

The problems of searching and mapping have been studied extensively in the robot navigation literature (for example, see [9, 13] for current surveys). Yet, there has been no previous work using competitive analysis for the problems that we consider. However, there have been some recent investigations, using competitive analysis, into some related problems that are set in unknown scenes. We now survey these investigations.

Papadimitriou and Yannakakis [10] pioneered the use of competitive analysis in the study of problems involving navigation of a scene using visual information. They consider the problem of finding a short obstacle avoiding path to a target with a *known* location in the plane. Papadimitriou and Yannakakis showed one can achieve a constant competitive factor if every obstacle in the scene has bounded aspect ratio. The *aspect ratio* of a convex polygonal object O is defined to be R/r where R is the radius of the smallest circle that circumscribes O and r is the largest circle that inscribes O [2]. The aspect ratio of a scene is defined as the maximum of the aspect ratio of any object in the scene. They also showed that no online algorithm with a constant competitive factor exists if the objects are convex polygons with unbounded aspect ratio.

Blum, Raghavan, and Schieber [2] continued studying the problem of finding an obstacle avoiding path to a known destination in the plane. They showed a tight bound of $\Theta(\sqrt{n})$ on the competitive factor in the case when the objects are oriented rectangles with width $\Omega(1)$, where n is the distance of the shortest obstacle avoiding path to the target. Blum *et al* also examined several other aspects of this problem, including allowing more general objects in some cases, randomization, the restriction to tactile robots, and generalization to higher dimensions. In addition, Karloff, Rabani and Ravid [8] give an $\omega(1)$ bound on the competitive factor achievable by randomized algorithms for this problem.

Baeza-Yates, Culberson, and Rawlins [1] consider several nice problems that, while not directly related to the problems that we consider, have a similar flavor to the searching problem. Deng and Papadimitriou [4] originally proposed using

competitive analysis to study the mapping problem. The offline version of mapping is sometimes called the watchman route problem (see for example [3]).

1.3. *Summary of results*

The ability of the robot to navigate quickly depends on the type of environment that it inhabits. One factor that makes it more difficult to navigate is the number of objects in the scene. Another property of the scene that makes searching hard is the aspect ratio of the objects [10, 2]. We will thus measure the competitive factor as a function of the number of objects and the aspect ratios of the objects. We say that a scene $\mathcal{S}$ is a $(k, \alpha, \bar{\alpha})$-scene if $\mathcal{S}$ consists of exactly k noncontiguous polygonal objects, with the largest aspect ratio of any of the k objects being α, and the average aspect ratio of these k objects being $\bar{\alpha}$. The average aspect ratio is just the sum of the aspect ratios of the k objects divided by k. In this paper we consider only convex polygonal objects to avoid mazelike scenes [2]. We define the function $\mathcal{M}(k, \bar{\alpha})$ to be $\min(k, \sqrt{k\bar{\alpha}})$.

In Section 2, we construct scenes where $\bar{\alpha} = \Theta(\alpha)$, and establish a bound of $\Omega(\mathcal{M}(k, \alpha))$, and hence $\Omega(\mathcal{M}(k, \bar{\alpha}))$, on the competitive factor for searching and mapping. To be more precise, let A be any online algorithm for these problems. Then there is a constant $c > 0$ such that for each k and each α there is a $(\Theta(k), \Theta(\alpha), \Theta(\bar{\alpha}))$-scene, on which the length of the tour generated using online algorithm A is at least $c\mathcal{M}(k, \bar{\alpha})$ times the length of the optimal obstacle avoiding path to complete the task. In Section 3, we analyze the competitive factor of the simple greedy heuristic, Nearest Neighbor (NN). For searching, the competitive factor for NN can be as high as $\Omega(2^k)$. For mapping we show that the competitive factor for NN is $\Omega(\alpha\mathcal{M}(k, \bar{\alpha}))$, and $O(\alpha \log k\, \mathcal{M}(k, \bar{\alpha}))$. The main reason that NN is not optimal for these problems is that the heuristic does not localize its search around the origin. In Section 4, we show that a careful modification to Nearest Neighbor, which we call Bifold Nearest Neighbor (BNN), has a competitive factor that is $\Theta(\log k\ \mathcal{M}(k, \bar{\alpha}))$ for mapping and searching. More importantly, BNN is almost as simple as Nearest Neighbor. Yet, the bound for BNN does not match the lower bound given in Section 2. Finally, we present a traversal based algorithm, Tourist, that achieves the optimal competitive factor of $\Theta(\mathcal{M}(k, \bar{\alpha}))$ for searching and mapping.

Independently of our investigation, Deng, Kameda and Papadimitriou [5] studied mapping in the case of arbitrary polygonal obstacles. Their main result was an O(1)-competitive algorithm for mapping the exterior, or interior, of a simple polygon. Their algorithm is $O(k)$-competitive algorithm for mapping a scene of k, not necessarily convex, polygonal objects. Thus our $O(\mathcal{M}(k, \bar{\alpha}))$ bound for mapping can be viewed as a refinement of their $O(k)$ bound for the case of convex polygonal objects. However, Deng et al. defined the mapping problem somewhat differently than we do. They not only require that the robot's path must see the boundary of each object, but also must see every point not

occupied by an object. While this additional condition does not burden the online algorithm, since it had to meet this condition anyway, it may greatly increase the cost incurred by the optimal offline algorithm. Thus while upper bounds on the competitive factor for one version of mapping apply to the other, the same is not true for lower bounds. We have a tight $\Omega(\mathcal{M}(k, \bar{\alpha}))$ bound on the competitive factor for our version of mapping, but for the version of Deng et al., the best lower bound for the competitive factor that is known for the case of convex polygonal objects is $\Omega(\sqrt{k})$ [2, 8].

2. Lower bounds

Theorem 2.1. *The competitive factor of any online algorithm for searching, or mapping, $(k, \alpha, \bar{\alpha})$-scenes is Ω $(\mathcal{M}(k, \alpha))$, and hence $\Omega(\mathcal{M}(k, \bar{\alpha}))$.*

Proof. Assume first that $\alpha < k$. We then prove an $\Omega(\sqrt{k\alpha})$ bound. We build a scene from the set of obstacles shown in Fig. 2a. For the rest of this proof we will call such a collection a *rectangle*. The critical fact is that by circling the rectangle the perimeter of each object can be seen, but the obstacle free square region in the center of the rectangle can not be seen. The width of the rectangle is 1 and the height is α. Notice that the aspect ratio of each polygonal object is $\Theta(\alpha)$. Also, observe that $\bar{\alpha} = \Theta(\alpha)$.

Let $r = \Theta(\sqrt{k\alpha})$ such that r/α is an integer. We then combine these obstacles as shown in Fig. 2b. Each rectangle in Fig. 2b represents the collection of objects from Fig. 2a. There are r/α rows and r columns. Hence the whole picture fits within a $2r$ by $2r$ square region. The dashed lines represent lines of sight through

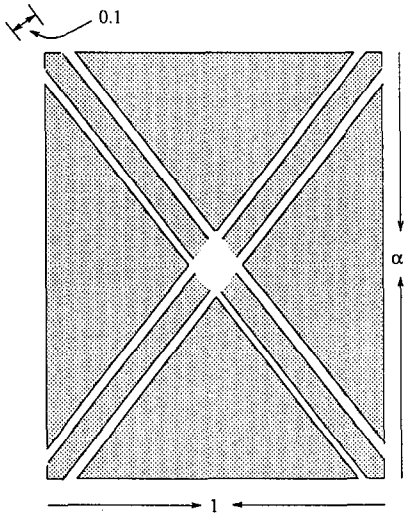

Fig. 2a.

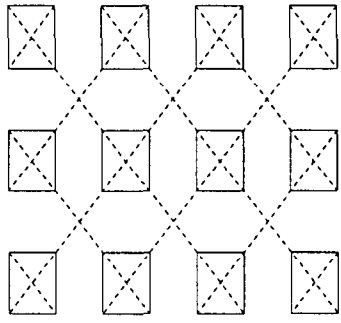

Fig. 2b.

the rectangles. One of the rectangles is then modified by adding a small square target to the obstacle free region in its center. The robot's goal in the search problem is to find this small square. To see this square the robot must visit the center of the rectangle.

The offline cost for searching is clearly $O(r)$. By walking around the outside of the scene, and by visiting the one modified rectangle, the offline algorithm for mapping can see the perimeters of all the objects while traveling at most a distance of $O(r)$. In the worst case the online algorithm must visit the center of each of the r^2/α rectangles, at a cost of $\Theta(\alpha)$ per visit. Thus the total cost to the robot is $\Omega(r^2)$. This gives us a lower bound on the competitive factor to be $\Omega(r)$, or equivalently $\Omega(\sqrt{k\alpha})$.

If $\alpha \geq k$ then $\mathcal{M}(k, \alpha) = k$. We can then repeat the above construction with the aspect ratio $\alpha = k$. □

In the case of scenes containing only squares, one can show a lower bound of $\Omega(\sqrt{k})$ on the competitive factor for searching and mapping [6]. Similarly, for three dimensional scenes containing cubes one can show a lower bound of $\Omega(k^{2/3})$ [6]. It should be clear from the proof of Theorem 2.1 that randomization does not provide any asymptotic benefit for the online algorithm.

3. Nearest Neighbor

In this section we analyze the competitive factor of the Nearest Neighbor (NN) heuristic. Given two points x and y in the scene, we define the distance between these points to be the length of the shortest obstacle avoiding path between x and y. We define the distance between two objects to be the minimum distance between any two points on their perimeters. That is, for two objects X and Y, let x and y be the points on the perimeters of X and Y, respectively, that minimize the distance between x and y. The distance between X and Y is then just the distance between x and y.

Nearest Neighbor. Select the closest object from the origin. Assume that X is the next object to be visited and that x_1 is the point on the perimeter of X that will be visited first. Starting from the point x_1, the object X will be circumnavigated. Let Y be the nearest unvisited object to X, and let x_2 and y_1 be the closest points on X and Y, respectively. From x_1 the robot traverses the shortest obstacle avoiding path to y_1 through x_2. (The condition that x_2 be an intermediate point is strictly unnecessary, but makes the analysis more transparent.) Note that Y is not necessarily the closest unvisited object to x_1. (However, all of our bounds still hold if the nearest unvisited object to x_1 was selected.) Notice that circumnavigating each visited obstacle guarantees that the robot can compute Y while at X. For searching, once NN has seen some part of the target it moves directly to

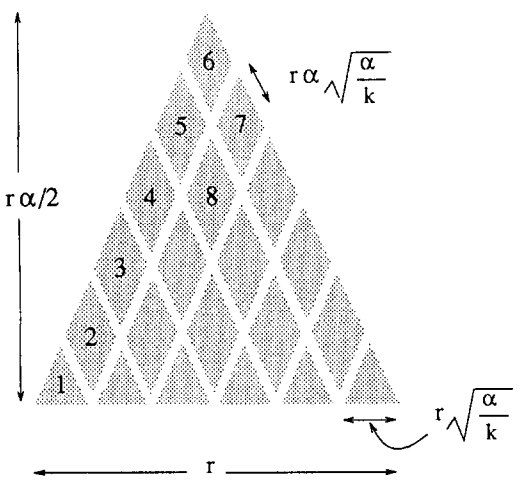

Fig. 3a.

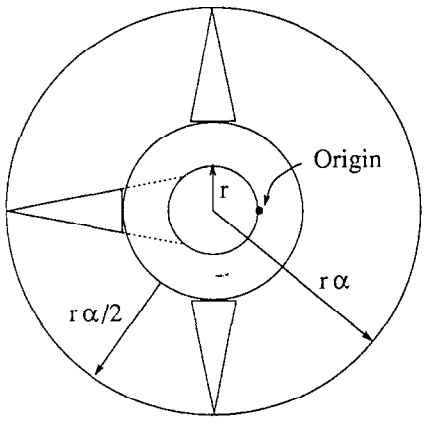

Fig. 3b.

the target and quits. For mapping, NN terminates once every point, not covered by an obstacle, is visible from some point on the robot's path.

Theorem 3.1. *Given any k and any α, there is $(\Theta(k), \Theta(\alpha), \Theta(\alpha))$-scene on which the competitive factor for Nearest Neighbor for mapping is at least $c\alpha\mathcal{M}(k, \alpha)$, for some constant c.*

Proof. As before, we construct a scene where $\bar{\alpha} = \Theta(\alpha)$. Assume for now that $\alpha < k$. We call the collection of objects pictured in Fig. 3a, a *triangle*. The scene in Fig. 3b contains $\Theta(\alpha)$ triangles placed in the left half of the area between the outer two circles. Also, the distance between any two triangles is at least r. The base of a triangle is a tangent to the middle circle. Each diamond in Fig. 3a has side length $r\alpha^{3/2}/\sqrt{k}$, and width $r\sqrt{\alpha/k}$. Furthermore, it is crucial that these obstacle free regions between the diamonds are narrow (say $\Theta(1/r)$). Each triangle contains $\Theta(k/\alpha)$ objects. The starting point of the robot is shown in Fig. 3b. It is selected such that, the nearest object is the corner object (numbered 1 in Fig. 3a) of the triangle directly below the starting point, assuming that ties are broken appropriately.

Observe that all the objects from each triangle can be seen by walking along the perimeter of the inner circle of radius r. Hence, the optimal offline cost is $O(r)$. On the other hand, when ties are broken appropriately, (see the numbering on Fig. 3a), NN will visit all the objects in at least $\Theta(\alpha)$ triangles. While visiting a triangle, NN will circumnavigate each object in the triangle before moving to another triangle. Thus, NN will visit $\Theta(k)$ objects with cost of $\Theta(r\alpha^{3/2}/\sqrt{k})$ per object, for a total cost of $\Theta(r\alpha^{3/2}\sqrt{k})$. Hence, we get a competitive factor of $\Theta(\alpha^{3/2}\sqrt{k})$.

If $\alpha \geq k$ then each triangle consists of a constant number of objects. Now Fig. 3b consists of $\Theta(k)$ triangles placed uniformly around the left half as shown in Fig. 3b. NN will visit $\Theta(k)$ objects at a cost of $\Theta(\alpha r)$ per object. Hence, we get a competitive factor of $\Omega(k\alpha)$. □

We now show that the heuristic NN achieves a competitive factor of $O(\alpha \log k\, \mathcal{M}(k, \bar{\alpha}))$. First we need several technical lemmas and definitions.

Lemma 3.1. *Let O be a convex polygonal object with aspect ratio α and area A and perimeter P. Then, $P^2 \leq 4\pi\alpha A$.*

Proof. Let O be inscribed by a circle c, with center o and radius r. Furthermore, let O be circumscribed by a concentric circle C, with radius at most $2\alpha r$ and at least αr, and with the property that C touches O at least one point x. Hence, $P^2 \leq 4\pi\alpha^2 r^2$. Consider the diameter of c perpendicular to the line xo. Let the endpoints of this diameter be t_1 and t_2. Then the triangle $\Delta t_1 x t_2$ is entirely contained within O by convexity. The area of $\Delta t_1 x t_2$, and hence the area of O, is at least αr^2. The result follows. □

Lemma 3.2. *Assume that a $(k, \alpha, \bar{\alpha})$-scene fits within an r by r square S. Then the sum of the perimeter of the objects is $O(r\mathcal{M}(k, \bar{\alpha}))$.*

Proof. Assume for now that $k \geq \bar{\alpha}$. Let P_i and A_i be the perimeter and the area of the ith object. Using Lemma 3.1 and Cauchy's inequality we have:

$$\left[\sum_{i=1}^{k} P_i\right]^2 \leq 4\pi\left[\sum_{i=1}^{k} \sqrt{\alpha_i A_i}\right]^2 \leq 4\pi\left[\sum_{i=1}^{k} \alpha_i\right]\left[\sum_{i=1}^{k} A_i\right] \leq 4\pi k\bar{\alpha}r^2$$

Therefore, $\sum_{i=1}^{i=k} P_i$ is at most $r\sqrt{4\pi\bar{\alpha}k} = 2\sqrt{\pi}\, r\mathcal{M}(k, \bar{\alpha})$. Now we consider the case when $k < \bar{\alpha}$. Due to convexity, an object inside S has a perimeter of at most $4r$. Therefore, the total cost of circumnavigating all of the objects is $4kr = 4r\mathcal{M}(k, \bar{\alpha})$. □

We now define what we call a *tourist graph* G of a $(k, \alpha, \bar{\alpha})$-scene contained in an r by r square S. G will be used to bound the optimal tour, and a slightly modified version of G will be used by the algorithm Tourist in the next section. We first define a collection $l(G)$ of lines segments that we later use to derive G. There is some ambiguity in the way that we form the set $l(G)$. But, for our results, any resultant $l(G)$ is satisfactory. The set $l(G)$ will be the union of several components, the first of which we call $l(G_1)$. Partition S into $\mathcal{M}(k, \bar{\alpha})^2$ boxes of side length $r/\mathcal{M}(k, \bar{\alpha})$. Consider the grid formed by the boxes. Let C be the set of objects that intersect some portion of the grid. The line segments in $l(G_1)$ are the portion of the sides of the boxes not overlapped by an object, and the sides of the objects in C. Notice that a side of a box, may be broken into many line segments.

We now consider objects that lie completely inside some box B. We form a set $l(B)$ of line segments that lie inside the box B. If an object X is completely within B then the sides of X are added as line segments to $l(B)$. Furthermore, if from some point x on the perimeter of X one can see some point p on a line segment in $l(G_1)$ that is inside or on the box B, then the line segment xp is added to $l(B)$. Only one line segment is added to $l(B)$ under this rule for each object in B. (Note that this is one place where ambiguity can arise in the construction.) If two objects, X and Y, have two points x and y on their perimeter, such that x and y are within B and x is visible from y, then we add the line segment xy to $l(B)$. Note that only one line segment between X and Y is added to $l(B)$ using this rule. (This is the other place where ambiguity can arise.) The set $l(G)$ is then the union of $l(G_1)$ and $l(B)$'s for each box B.

Given a collection of line segments $l(G)$, the corresponding graph G is constructed in the following manner. Each endpoint of a line segment is a vertex. Further, if some line segment xy has an endpoint, say x, on another line segment vw then x becomes a vertex and the line segment vw is replaced by the two line segments vx and wx. Notice that if two line segments intersect at some point p, which is on the interior of both line segments, then p is not a vertex. The line segments then form the edges of the graph. Similarly, we can form a graph G_1 from the set $l(G_1)$ line segments. Fig. 4 shows a G_1 graph for a sample scene. Notice that both G and G_1 are connected.

Lemma 3.3. *Assume that a $(k, \alpha, \bar{\alpha})$-scene fits within an r by r square S. Then there is a tour of the scene with total length at most $cr\mathcal{M}(k, \bar{\alpha})$, where c is some constant.*

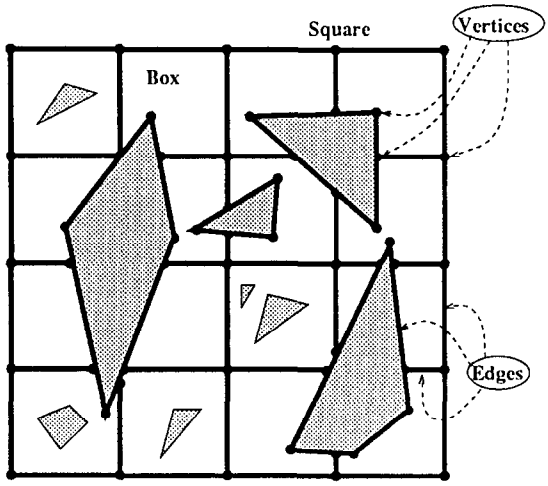

Fig. 4. The graph G_1.

Proof. Consider a depth first traversal of a tourist graph G of the scene that completely traverses an object when it is first encountered, and that does not traverse an edge if it is not the side of some object and the other endpoint has been visited. By Lemma 3.2 the total length of the edges derived from the perimeters of objects is $O(r\mathcal{M}(k, \bar{\alpha}))$. The total length of the edges derived from the grid edges is at most $2r(\mathcal{M}(k, \bar{\alpha}) + 1)$. Hence, it suffices to show that the total length of the at most $2k$ nonperimeter edges is $O(r\mathcal{M}(k, \bar{\alpha}))$. Each time any such edge was traversed the robot was either visiting an object for the first time, or just leaving a visited object. The length of any edge in $G - G_1$ is at most $\sqrt{2}\, r/\mathcal{M}(k, \bar{\alpha})$, so we need only verify that $\sqrt{2}\, kr/\mathcal{M}(k, \bar{\alpha}) = O(r\mathcal{M}(k, \bar{\alpha}))$. This follows since $k \leq \mathcal{M}(k, \bar{\alpha})^2$. □

The proof of the following lemma is similar to the proof that Nearest Neighbor has a competitive factor of $O(\log n)$ in a metric space [12].

Lemma 3.4. *Assume that a $(k, \alpha, \bar{\alpha})$-scene $\mathcal{S}$ fits within an r by r square S. Then the total distance traveled by Nearest Neighbor, starting from the center of S, is* $O(r \log k\, \mathcal{M}(k, \bar{\alpha}))$.

Proof. While an object X is visited for the first time, let x_1 (respectively x_2) be the first (respectively last) point on the perimeter of the object visited by NN. Note that NN walks along the perimeter of X and then selects the next closest object (say Y) to visit. So the distance traveled between x_1 and x_2 is at most 3/2 times the perimeter of X. By Lemma 3.2, the total distance traveled by NN while circumnavigating objects is $O(r\mathcal{M}(k, \bar{\alpha}))$.

We now deal with the distance traveled by NN when moving from one object to another. Assume that after visiting an object X, NN visited the object Y. Then let $\mathrm{NN}(X)$ be the length of the shortest obstacle avoiding path P from any point (say x_2) on X to any point (say y_1) on Y. Let $C = \{C_1, C_2, \ldots, C_n\}$ be a collection of n objects. Initially, $n = k$ and C is the set of all objects in the scene. Let $\mathrm{OPT}(C)$ be the length of the optimal tour to visit and circumnavigate just the objects in C in the scene starting from the origin. Without loss of generality, let us assume that the optimal tour visits objects in C in the order, $C_1, C_2, \ldots, C_n$. Consider a pair of objects C_i and C_{i+1}. Let $\mathrm{OPT}(C_i, C_{i+1})$ be the distance between objects C_i and C_{i+1}. Assume for the moment that NN visits C_i before C_{i+1}. Then by the definition of NN, $\mathrm{NN}(C_i) \leq \mathrm{OPT}(C_i, C_{i+1})$. Similarly, if NN visits C_{i+1} before C_i then $\mathrm{NN}(C_{i+1}) \leq \mathrm{OPT}(C_i, C_{i+1})$. Hence,

$$\min(\mathrm{NN}(C_i), \mathrm{NN}(C_{i+1})) \leq \mathrm{OPT}(C_i, C_{i+1}).$$

Therefore, there is a collection $D \subseteq C$, of at least $\lfloor n/2 \rfloor$ objects, such that $\sum_{O \in D} \mathrm{NN}(O) \leq \mathrm{OPT}(C)$. We then let $C = C - D$ and repeat this argument. Note the following. Since, the initial cardinality of C is k, this process is repeated at most $O(\log k)$ times until the number of objects in C is 1. For any object $X \in \mathcal{S}$,

$\mathrm{NN}(X) \leq \mathrm{OPT}(\mathcal{S})$. Since objects are removed from C, the values $\mathrm{OPT}(C)$ form a monotonically decreasing sequence. Therefore, by applying lemma 3.3, we get

$$\sum_{O \in \mathcal{S}} \mathrm{NN}(O) = \mathrm{O}(r \log k\mathcal{M}(k, \bar{\alpha})). \qquad \square$$

The following lemma helps us to establish a lower bound on the offline cost for mapping (or searching) if the scene contains objects with aspect ratio at most α.

Lemma 3.5. *Let α be the aspect ratio of a convex polygonal object, and let θ be the smallest internal angle of the object. If $\theta \leq \pi/2$ then* $\tan(\theta/2) \geq 1/(4\alpha)$.

Proof. Let the object be inscribed by a circle c, with center o and radius r. Furthermore, let the object be circumscribed by a concentric circle C, with radius at most $2\alpha r$ and at least αr. Let a, b, and c be three consecutive corners of the convex polygonal object such that $\angle abc = \theta$ is the smallest internal angle. Draw two tangents bt_1 and bt_2 to the circle c from b where t_1 and t_2 are the points on c touched by two tangents respectively. Observe that t_1 and t_2 are not outside the object. Therefore, $\angle t_1 b t_2 = \theta_1 \leq \theta$. Observe that $\angle t_1 bo = \theta_1/2$ and therefore $\tan(\theta_1/2) = r/d(t_1, b)$. Since t_1 and b are inside the circle C, $d(t_1, b) \leq 4\alpha r$. Therefore, $\tan(\theta_1/2) \geq 1/(4\alpha)$. Observe that the tan function in the range 0 to $\pi/2$ is increasing. Therefore, $\tan(\theta/2) \geq \tan(\theta_1/2) \geq 1/(4\alpha)$. $\square$

We are now ready to determine the competitive factor of NN for mapping.

Theorem 3.2. *The competitive factor of the online algorithm NN for mapping $(k, \alpha, \bar{\alpha})$-scenes is* $\mathrm{O}(\alpha \log k\, \mathcal{M}(k, \bar{\alpha}))$.

Proof. Let r be the side length of the smallest square that contains all of the objects in the terrain. We first show that the offline cost will be $\Omega(r/\alpha)$. Of all the points on the perimeter of objects, let p be the furthest from the origin o, using the obstacle avoiding distance as the measure. Note that we can assume, without loss of generality, that p is a corner of some object. The length of the line segment op is at least $r/2$. Let a and b be the two adjacent corners to the corner p. Let $\angle apb = \theta$. By Lemma 3.5, we know that $\tan\theta = \Omega(1/\alpha)$. Without loss of generality, assume that $\angle apo \geq \theta/2$. In order to see the line segment ap, the robot has to move a distance of at least $(r/2)\sin(\theta/2)$. If $\theta \geq \pi/2$, the offline cost is $\Omega(r)$, and the result follows. Otherwise, $(r/2)\sin(\theta/2) \geq (r/2)\tan(\theta/2) \geq r/(8\alpha)$. We finish the proof by noting that, by Lemma 3.4, the cost of NN is $\mathrm{O}(r \log k\, \mathcal{M}(k, \bar{\alpha}))$. $\square$

In contrast, for searching the competitive factor for NN can be exponential in the number of objects.

Theorem 3.3. *There is a $(k, \alpha, \bar{\alpha})$-scene such that the competitive factor for searching using the algorithm Nearest Neighbor in the plane is* $\Omega(2^k)$.

Proof. Assume the robot starts at the origin. Objects (for example, unit squares) are placed at the points $(-1, 0)$, $(1, 0)$, $(3, 0)$, $(7, 0), \ldots, (2^i - 1, 0) \cdots (2^{k-1} - 1, 0)$. A very small target (say a tiny square) is placed just to the left of the object at $(-1, 0)$ so that it can not be seen from any point on the other objects. NN will move first to the object at $(1, 0)$ then to the object at $(3, 0)$ etc., incurring a cost of $\Theta(2^k)$, before returning to the object at $(-1, 0)$ and seeing the target. □

4. Bifold algorithms

In this section we first present Bifold Nearest Neighbor (BNN), a modification of Nearest Neighbor that has a competitive factor that is only a $O(\log k)$ factor away from optimal. We then present the algorithm Tourist, which has an asymptotically optimal competitive factor.

Throughout the description let $\mathcal{S}$ be an $(k, \alpha, \bar{\alpha})$-scene. Assume that the robot starts at the origin. Let $\rho(0)$ be the distance to the closest object, and let $\rho(i) = 2\rho(i-1)$. Denote by S_i the axis parallel square centered at the origin that has side length $\rho(i)$. Let R_i be the region reachable from the origin without leaving S_i. Denote the number of objects partially or fully within S_i by k_i, denote the average aspect ratio of those k_i objects by $\bar{\alpha}_i$. Observe that $\mathcal{M}(k_i, \bar{\alpha}_i)$ is monotonically non-decreasing with respect to i. This is because k_i and $k_i\bar{\alpha}_i$, the sum of the aspect ratios of the objects, are monotonically nondecreasing with respect to i.

Bifold Nearest Neighbor. BNN is divided into phases. In the ith phase BNN visits obstacles in R_i. When BNN first visits an object X, it travels along the entire portion of the perimeter of X that is within R_i. Assume that BNN has just accomplished this. The robot then selects, as the next object Y to visit, the one closest to X that satisfies the following: Y has not been visited in this phase, and some part of Y lies within R_i. For searching, phase i ends when all objects in R_i are visited, and BNN terminates after moving to the target when it is is first seen. For mapping, BNN traverses the perimeter of R_i before the end of the phase, and then terminates if every point, not covered by an obstacle, has been seen from the current tour.

Lemma 4.1. *The cost to BNN for phase i is bounded from above by $c\rho(i)\log k_i\, \mathcal{M}(k_i, \bar{\alpha}_i)$, for some constant c.*

Proof. Analogous to the proof of Lemma 3.4. □

Theorem 4.1. *The competitive factor of Bifold Nearest Neighbor for mapping (and searching) $(k, \alpha, \bar{\alpha})$-scenes is $O(\log k\, \mathcal{M}(k, \bar{\alpha}))$.*

Proof. The proof is by induction on the phases. The induction hypothesis is: The total cost of BNN at the end of phase i is bounded from above by $2c\ \rho(i)\log k_i\ \mathcal{M}(k_i, \bar{\alpha}_i)$. Note that the constant c is from Lemma 4.1.

Assume that the induction hypothesis is true for the first $i-1$ phases. By Lemma 4.1, the cost incurred by the robot during the phase i is at most $c\rho(i)\log k_i\ \mathcal{M}(k_i, \bar{\alpha}_i)$. By the induction hypothesis the total cost for the first $i-1$ phases is at most

$$2c\rho(i-1)\log k_{i-1}\mathcal{M}(k_{i-1}, \bar{\alpha}_{i-1}).$$

The induction hypothesis for i then follows since $2\rho(i-1)=\rho(i)$, $k_{i-1}\leq k_i$ and $\mathcal{M}(k_{i-1}, \bar{\alpha}_{i-1})\leq \mathcal{M}(k_i, \bar{\alpha}_i)$. The result for searching then follows since the offline cost is at least $\rho(i-1)$.

For mapping let r be the distance of the furthest point reached by the optimal offline algorithm, and for searching let r be the distance to the target. Then, in both problems, BNN terminates at by the end of the first stage for which $\rho(i)\geq r$. □

For ease of explanation, we describe several versions of the Tourist algorithm, with each successive version handling more complicated scenarios of searching and mapping. Tourist intuitively attempts to conduct a depth first search of the tourist graph of the scene.

Tourist1. Tourist1 accepts two parameters M, and r, and has a locally defined constant c. Let S be the r by r square centered at the origin. Tourist1 divides S into M^2 boxes of size r/M by r/M. Let $l(G_1)$ be the set of line segments induced by the portion of the object sides, and the portion of the box sides, that are reachable from the origin by an obstacle avoiding path that does not leave S. Let G_1 be derived from $l(G_1)$ as in Section 3. Tourist1 first conducts a depth first search of G_1. Note that *a priori* knowledge of G_1 is not needed to accomplish this. While traversing G_1 the location of each portion of an object, which is completely within a box, that Tourist1 sees during this traversal is stored in a stack. After finishing with G_1, Tourist1 will repeat the traversal of G_1 with the intention of visiting those objects that are completely inside some box. Suppose that during the second traversal of G_1, Tourist1 is at a point p on the perimeter of a box B (or inside B) from which it saw a portion of unvisited object X. Tourist1 then leaves its retraversal of G_1 temporarily, circumnavigates X, and removes all instances of X from the stack. While circumnavigating X, Tourist1 might see portions of unvisited objects in B. Tourist1 then puts these objects in the stack, and visits them recursively. After visiting all objects added to the stack after leaving the point p, Tourist1 returns to conducting its retraversal of G_1 from p. If at some point Tourist1 travels a distance of more than crM then it returns to the origin, and returns to the calling procedure that it was unsuccessful. One can see as a simple corollary to Lemma 3.3 that there is a constant c such that if

$\mathcal{M}(k, \bar{\alpha}) \leq M \leq 2\mathcal{M}(k, \bar{\alpha})$ Tourist1 will successfully visit each object in S. This c is the constant used in Tourist1.

```
function Tourist1 (r, M:integer):Boolean;
const c
begin
  Perform a traversal the tourist graph
  until a total distance of crM is traveled or the tour of G is complete
  if the tour of G was completed then
    return Tourist := true
    else return Tourist1 := false
end;
```

Lemma 4.2. *Let $\mathcal{S}$ be some $(k, \alpha, \bar{\alpha})$-scene with each object contained at least in part in the r by r square S centered at the origin. If $\mathcal{M}(k, \bar{\alpha}) \leq M \leq 2\mathcal{M}(k, \bar{\alpha})$ then Tourist1 will fully complete its traversal of a tourist graph of the scene relative to square S and the length of the tour is at most crM.*

Proof. The proof is then analogous to the argument found in the proof of Lemma 3.3. □

Tourist2. Tourist2 accepts one parameter r. Tourist2 sets $M = 1$, and executes Tourist1 accordingly. If Tourist1 returns that it was unsuccessful then Tourist2 doubles M, and restarts Tourist1 with this new value of M. Lemma 4.2 guarantees that Tourist2 terminates. In mapping, if some points are still not visible from the online path constructed so far, and in searching, if the target has not been seen (or visited), then Tourist2 returns that its task is not complete.

```
function Tourist2 (r:integer):Boolean;
begin
  M := 1;
  while not(Tourist1(r, M)) do M := 2M
  if the task is not complete then return false
  else return true
end;
```

Lemma 4.3. *The total distance traveled while executing Tourist2 is at most $8cr\mathcal{M}(k, \bar{\alpha})$, where r is the parameter of Tourist2, and c is the constant defined in the function Tourist1.*

Proof. The proof is by induction on number of calls to Tourist1. Our induction hypothesis is: At the end of the ith call, distance traveled by Tourist2 is at most $4cr\mathcal{M}(k, \bar{\alpha})_i$, where $\mathcal{M}(k, \bar{\alpha})_i = 2^{i-1}$ is the guessed value of $\mathcal{M}(k, \bar{\alpha})$.

The basis case where $i = 0$ is trivial. The distance traveled in the $(i+1)$st call of Tourist1 is at most $2cr\mathcal{M}(k, \bar{\alpha})_{i+1}$. Since Tourist1 returns to the center, the factor 2 appears in the bound. By induction hypothesis, the total distance traveled by Tourist2 during the first i calls of Tourist1 is at most $4cr\mathcal{M}(k, \bar{\alpha})_i$. Notice that $2\mathcal{M}(k, \bar{\alpha})_i = \mathcal{M}(k, \bar{\alpha})_{i+1}$. Therefore, total distance traveled by Tourist2 during the first $i+1$ calls is at most $4cr\mathcal{M}(k, \bar{\alpha})_i$. This proves the induction hypothesis. By Lemma 4.2, Tourist2 terminates after i calls to Tourist1 when $\mathcal{M}(k, \bar{\alpha}) \leq \mathcal{M}(k, \bar{\alpha})_i \leq 2\mathcal{M}(k, \bar{\alpha})$. The result follows. □

Tourist (final version)**.** The final version of Tourist is then obtained by applying Tourist2 to the exponentially growing neighborhoods S_i in a manner analogous to how BNN was obtained from NN.

```
procedure Tourist;
begin
  ρ(1):=the distance to the closest object;
  i=1
  while not Tourist2(ρ(i)) do
    begin
      i:=i+1
      ρ(i):=2ρ(i-1)
    end
end;
```

Lemma 4.4. *Given a* $(k, \alpha, \bar{\alpha})$*-scene, the total distance traveled by Tourist, say* P_i*, is at most* $16c\rho(i)\mathcal{M}(k, \bar{\alpha})$*, where i is the number of calls to Tourist*2 *and where c is the constant from the function Tourist*1*.*

Proof. The proof is by induction on number of calls to Tourist2. The distance P_i is P_{i-1} plus the distance traveled during the ith call of Tourist2. By Lemma 4.3, the latter cost is at most $8c\rho(i)\mathcal{M}(k, \bar{\alpha})$. The result follows since

$$\begin{aligned} P_{i-1} + 8c\rho(i)\mathcal{M}(k, \bar{\alpha}) &\leq 16c\rho(i-1)\mathcal{M}(k, \bar{\alpha}) + 8c\rho(i)\mathcal{M}(k, \bar{\alpha}) \\ &\leq 16c\rho(i)\mathcal{M}(k, \bar{\alpha}) \qquad \square \end{aligned}$$

Theorem 4.2. *The competitive factor of Tourist for mapping (and searching)* $(k, \alpha, \bar{\alpha})$*-scenes is* $O(\mathcal{M}(k, \bar{\alpha}))$*.*

Proof. By an analogous argument to the one found in proof of Theorem 4.1. □

5. Open problems

It is interesting to note that at the heart of the searching and mapping problems is a graph traversal game, which we will call *Hide and Seek*. In Hide and Seek

both the seeker, say a cat, and the hider, say a mouse, start at the same position in some space containing k havens for the mouse. After the mouse hides in one of k havens the cat searches the possible havens looking for the one hiding the mouse. The cat's goal is to minimize the ratio of the distance that it travels over the shortest path for the mouse. The competitive factor for Hide and Seek in d-dimensional Euclidean space is $\Theta(k^{(d-1)/d})$ [6]. Hide and Seek is similar Raghavan and Snir's Cat and Mouse Game [11].

There is a number of obvious ways in which the results in this paper could be extended. There is $\Theta(\log k)$ gap between the lower and upper bounds of Bifold Nearest Neighbor. It would also be interesting to consider extending the results for searching and mapping to nonconvex objects. To correctly capture the hindrance posed by nonconvex objects, or three dimensional objects, one seems to need to introduce a different notion of aspect ratio, which we call resistivity (see [6]). An object O is α resistive if for every pair of points x and y on the boundary of O the shortest path from x to y along the boundary of O is at most α times the straight line distance between x and y. Resistivity and aspect ratio are asymptotically equivalent for convex two-dimensional objects.

It would be interesting to consider a generalization of searching in which we allow multiple robots to cooperatively search the scene. Here, instead of measuring the distance traveled by the robots, we measure time taken to complete the task. Since this does not benefit the adversary we would like to devise a strategy to improve the competitive factor by $O(m)$, where m is the number of robots. Some preliminary results can be found in [6]. One could also consider the problem of finding a short path to a *known* destination in the case of multiple robots.

One could also assume that a robot must be close to an object to fully map the object. This is perhaps a more realistic assumption in many situations. We call the resulting problem the *visual traveling salesman problem* since the robot must then essentially visit each object. We give a constant competitive algorithm for this problem in [7]. This shows that the reason that the competitive factors are so high for mapping is the adversary's ability to map from a distance.

References

[1] R. Baeza-Yates, J. Culberson and G. Rawlins, Searching in the plane, Inform. Comput, to appear.

[2] A. Blum, P. Raghavan and B. Schieber, Navigating in unfamiliar geometric terrain, Proceedings of the Twenty-Third Annual ACM Symposium of Theory of Computing (1991) 494–504.

[3] W. Chin and S. Ntafos, Optimum Watchman routes, Inform. Process. Lett. 28 (1988) 39–44.

[4] X. Deng and C. Papadimitriou, Exploring an unknown graph, Proceedings of the Thirty-First Annual Symposium on Foundations of Computer Science (1990) 355–361.

[5] X. Deng, T. Kameda and C. Papadimitriou, How to learn an unknown environment, Proceedings of the Thirty-Second Annual Symposium on Foundations of Computer Science (1991) 298–303.

[6] B. Kalyanasundaram and K. Pruhs, A competitive analysis of nearest neighbor based algorithms for searching unknown scenes, Proceedings of Ninth Annual Symposium on Theoretical Aspects of Computer Science (1992) 147–157.

[7] B. Kalyanasundaram and K. Pruhs, Constructing competitive tours from local information, Technical Report, Computer Science Department, University of Pittsburgh, 1992.

[8] H. Karloff, Y. Rabani and Y. Ravid, Lower bounds for randomized k-server and motion planning algorithms, Proceedings of the Twenty-Third Annual ACM Symposium of Theory of Computing (1991) 278–288.

[9] V. Lumelsky, S. Mukhopadhyay and K. Sun, Dynamic path planning in sensor-based terrain acquisition, IEEE Trans. Robot. Automat. 6 (1990) 462–472.

[10] C. Papadimitriou and M. Yannakakis, Shortest paths without a map, Proceedings of the Sixteenth Annual International Colloquium on Automata, Languages, and Programming (1989) 610–620.

[11] P. Raghavan and M. Snir, Memory vs. Randomization in on-line algorithms, Proceedings of the Sixteenth Annual International Colloquium on Automata, Languages, and Programming, (1989) 687–703.

[12] D. Rosenkrantz, R. Stearns, and P. Lewis, An analysis of several heuristics for the traveling salesman problem, SIAM J. Comput. 6 (1977) 563–581.

[13] J. Schwartz and C. Yap, Algorithmic and Geometric Aspects of Robots (Lawrence Erlbaum, London).